

Unsupervised learning of image motion by recomposing sequences

Andrew Jaegle*, Stephen Phillips*, Daphne Ippolito, and Kostas Daniilidis

University of Pennsylvania

Philadelphia, PA USA

{ajaegle, stephi, daphnei, kostas}@seas.upenn.edu

Abstract

We propose a new method for learning a representation of image motion in an unsupervised fashion. We do so by learning an image sequence embedding that respects associativity and invertibility properties of composed sequences with known temporal order. This procedure makes minimal assumptions about scene content, and the resulting networks learn to exploit rigid and non-rigid motion cues. We show that a deep neural network trained to respect these constraints implicitly identifies the characteristic motion patterns of many different sequence types.

Our network architecture consists of a CNN followed by an LSTM and is structured to learn motion representations over sequences of arbitrary length. We demonstrate that a network trained using our unsupervised procedure on real-world sequences of human actions and vehicle motion can capture semantic regions corresponding to the motion in the scene, and not merely image-level differences, without requiring any motion labels. Furthermore, we present results that suggest our method can be used to extract information useful for independent motion tracking, localization, and nearest neighbor identification. Our results suggest that this representation may be useful for motion-related tasks where explicit labels are often very difficult to obtain.

1. Introduction

Visual motion perception is a key component of ethological and computer vision. By understanding how a sequence of images reflects an agent’s motion and the motion of objects in the world around it, the agent can better judge how to act in that setting. For example, a fly can use visual motion cues to dodge a rapidly approaching hand and to distinguish this threat from the motion generated by a looming landing surface [1]. In computer vision, motion is an important cue for understanding the nature of actions, disambiguating the 3D structure of a scene, and predicting future scene structure. Because of the importance of motion cues for interpreting and navigating the visual world, it has

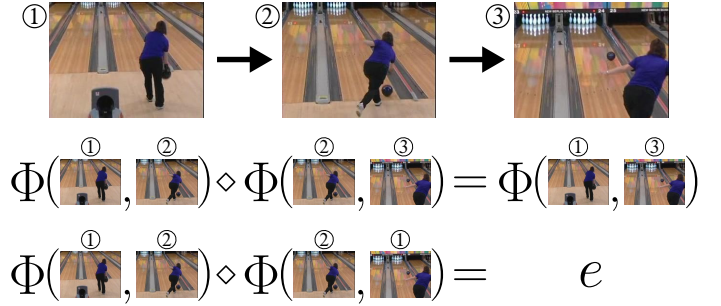


Figure 1: The goal of this work is to learn to represent image motion by capturing the group properties of associativity and invertibility, which are also properties of the motion of real-world scenes. By exploiting these properties, we learn to represent image motion without labels.

been extensively studied from computational, ethological, and biological perspectives.

In computer vision, the problem of motion representation has typically been approached from either a local or global perspective. Local representations of motion are exemplified by optical flow and its relatives. Flow represents image motion as the 2D displacement of individual pixels of an image, giving rich low-level detail while foregoing a compact representation of the underlying structure of the flow field. In contrast, global representations such as those used in odometry and simultaneous localization and mapping (SLAM) can coherently explain the movement of the whole scene in a video with relatively few parameters using a representation such as a point cloud and a set of poses. However, such representations typically rely on a rigid world assumption and ignore non-rigid elements and independent motions, thus discarding useful motion information and limiting their applicability.

For an agent to understand and act based on scene motion, it is important that the agent have access to a representation that is global in the scene content and flexible enough to accommodate scene non-rigidities and motion due to both itself and others. We propose to learn a motion

* Authors contributed equally.

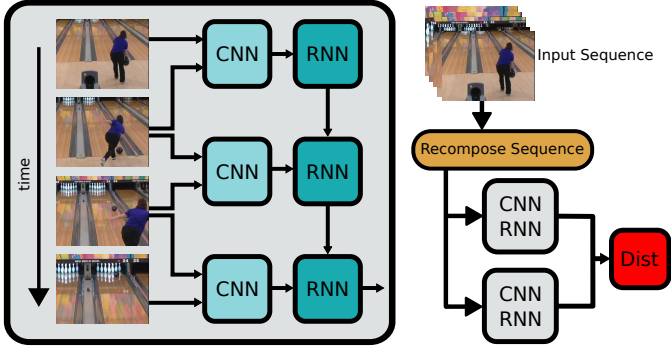


Figure 2: Schematic network architecture. Given a sequence, the CNN computes a representation over pairs of frames. The RNN takes the sequence of CNN outputs as input. The RNN output at the final step of the sequence is treated as the sequence embedding. During training, the distance between two sequence embeddings is used as a loss: representations are pushed together for equivalent sequences and apart for inequivalent sequences. Training encourages the representation to enforce the desired properties of associativity and invertibility.

representation with both of these properties.

However, supervised training of such a representation is challenging: explicit motion labels are difficult and expensive to obtain, especially for nonrigid scenes where it is often unclear how the structure and motion of the scene should be decomposed. Accordingly, we propose to learn this representation in an unsupervised fashion. To do so, we formulate a representation of visual motion that respects associations and inversions of temporal image sequences of arbitrarily length (Figure 4). We describe how these properties can be used to train a deep neural network to represent visual motion in a low-dimensional, global fashion.

Our contributions are as follows:

- We describe a new procedure for unsupervised learning of global motion by encouraging representational consistency under association and inversion given valid sequence compositions.
- We present evidence that the learned representation captures the global structure of motion in many settings without relying on hard-coded assumptions about the scene or by explicit feature matching.
- We show that our method can learn global representations of motion when trained on real-world datasets such as HMDB51 and KITTI tracking.

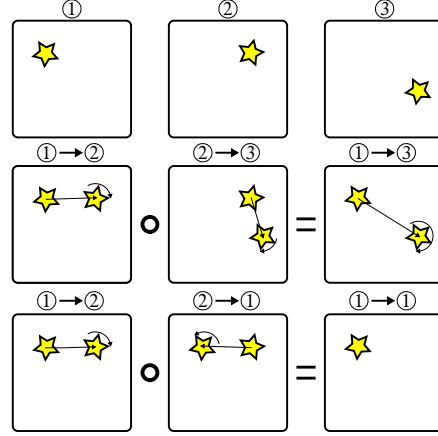


Figure 3: A simple illustration of the group properties of motion. Here, a rigid shape translates and rotates for three frames. Row 1: the world state in the three frames. Row 2: associativity - composing the motion between the first and the second frames with the motion between the second and third frames produces the motion between the first and third frames. Row 3: invertibility - composing the motion between the first and second frames with the motion between the second and first frames produces a null motion. As discussed in the text, the same properties hold for motion in more general scenes.

2. Related Work

2.1. Motion representations

Much of the work in visual odometry, structure from motion (SfM), and simultaneous localization and mapping (SLAM) uses a global, rigid representation of motion [2]. Here, motion is represented as a sequence of poses in $\mathcal{SE}(3)$ perhaps along with a static point cloud, both of which are refined when further visual measurements are obtained [3], [4]. To handle violations of the rigidity assumption, outlier rejection schemes are typically used to remove incongruous features and improve estimates [5] [4] [6]. These approaches have achieved great success in many applications in recent years, but they are unable to represent non-rigid or independent motions.

On the other end of the spectrum, several local, non-rigid representations are commonly used in computer vision. The most common of these is optical flow, which estimates pixel-wise motion over the image, typically constraining it with a smoothness prior [7]. Optical flow encodes 2D image motion only, and hence is limited in its capacity to represent general 3D motions. Scene flow [8] and non-rigid structure from motion [9] represent a larger class of 3D motions by generalizing optical flow to the estima-

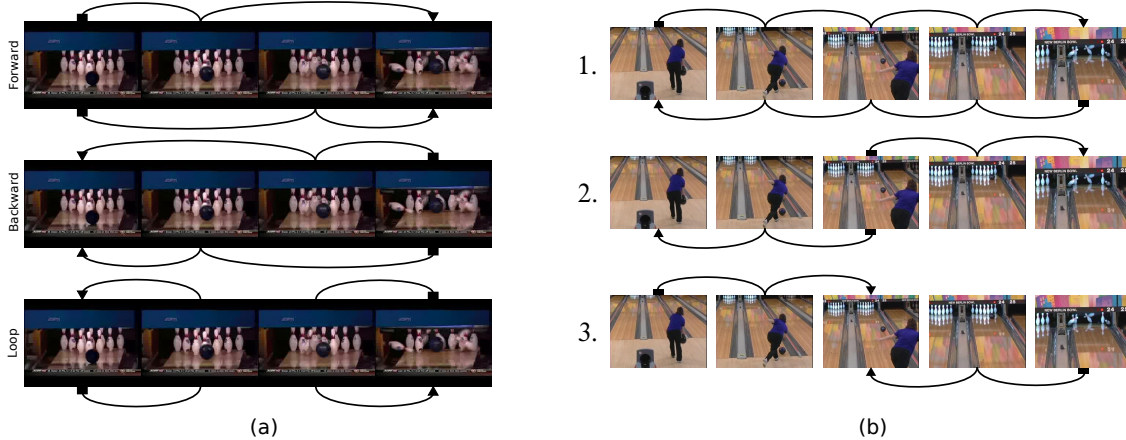


Figure 4: (a) We sample frames so as to enforce associativity and invertibility in recomposed sequences. Sequences of the same type (e.g. two forward sequences) are used as positive examples during training, while sequences of different types (e.g. a forward and a backward sequence) are used as negative examples. During training, sequences of varying length are used. (b) To prevent the network from learning representations of motion that only match the first or last frame of a sequence, we sample sequences using three different sampling configurations. Forward and backward sequences either have the same (row 2) or different starting frames (rows 1,3) and are drawn from either the same subsequence (row 1) or from temporally adjacent subsequences (rows 2,3).

tion of 3D point trajectories (dense or sparse, respectively). However, optical flow, scene flow, and nonrigid structure from motion represent motion only at local regions (typically single points).

Philosophically most similar to our approach is work designing or learning spatio-temporal features (STFs) [10]. STFs are localized and maintain the flexibility to represent non-rigid motions, but they typically include a dimensionality reduction step and hence are more global in purview than punctate local representations, such as optical flow. More recent work has used convolutional neural nets (CNNs) to learn task-related STFs directly from images, including [11] and [12]. Unlike our work, both of these approaches are restricted to fixed temporal windows of representation.

2.2. Unsupervised learning

In computer vision, unsupervised learning often refers to methods that do not use explicit, ground-truth labels but instead use “soft” labels derived from the structure of images as a learning signal. Our work is unsupervised in this sense. Recent work has used knowledge about the geometric or spatial structure of natural images to train representations of image content in this way. For example, [13] and [14] exploit the relationship between the location of a patch in an image and the image’s semantic content. Both works train a CNN to classify the correct configuration of image patches, and show that the resulting representation can be fine-tuned for image classification. Other work has ex-

ploited well-known geometric properties of images to learn representations. For example, [15] and [16] train networks to estimate optical flow using the brightness constancy assumption and a smoothness prior as a learning signal, while [17] uses the relationship between depth and disparity to learn to estimate depth from a rectified stereo camera pair with a known baseline.

In our work, we use the group properties of motion as the basis for learning to represent motion. To our knowledge, we are the first to exploit these properties for unsupervised representation learning. Other groups have used temporal information as a signal to learn representations. However, these representations are typically of static image content rather than motion. One approach is to learn representations that are stable between temporally adjacent frames [18]. This approach has its origins in slow feature analysis [19] and is motivated by the notion that latents that vary slowly are often behaviorally relevant. More similar to the present work is [20], which shuffles the order of images to learn representations of image content. Unlike our work, their approach is designed to capture single image properties that can be correlated with temporal order rather than motion itself. Their shuffling procedure does not respect the properties that form the basis of our learning technique. Other works exploring unsupervised learning from sequences include [21], which learns a representation equivariant to the egomotion of the camera, and [22], which learns to represent images by explicitly regressing the egomotion between

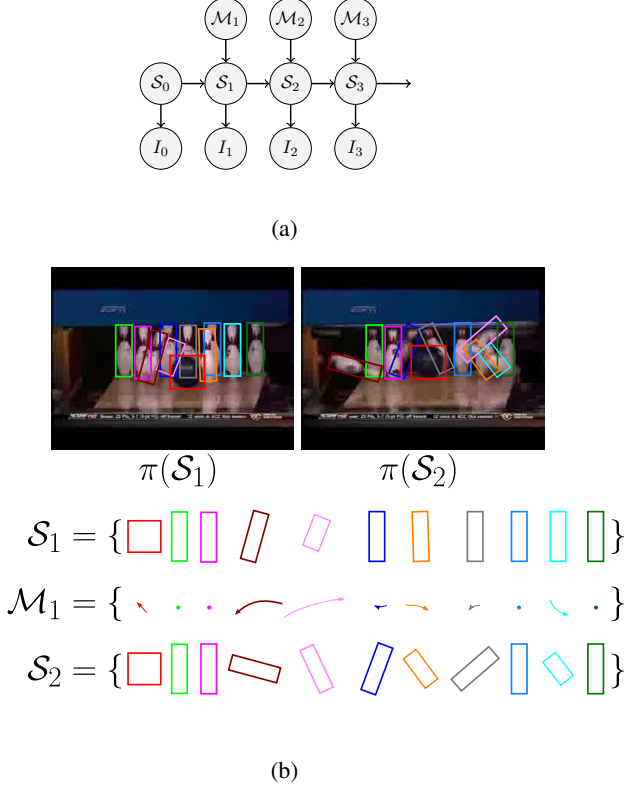


Figure 5: (a) Graphical model of the relationship between the latent structure and motion and the observed images of a sequence. We attempt to represent the motion \mathcal{M}_t sequence from the observed image sequence I_t with a learned model of motion. (b) In this view, a scene’s structure at time t , $S_t = \{S_t^1, \dots, S_t^{11}\}$ consists of 11 independent objects (here: the bowling ball and pins). The motion M_t describes the transformation between the structure at times t and $t+1$. In this case, $M \in \mathcal{SE}(3)^{11}$. We do not specify the scene decomposition or the motion group: the model must infer these during learning.

two frames. However, neither of these papers attempt to learn to represent motion; instead, they use motion as a learning cue.

3. Approach

3.1. Group properties of motion

We base our method on the observation that the latent motion in image sequences respects associativity and invertibility. Associativity refers to the fact that composing the motions of the components of a sequence gives the motion of the whole sequence. Invertibility refers to the fact that composing the motion between a sequence and the reversed sequence results in a null motion, as illustrated in Figure 3. To see that a latent motion space respects these

properties, first consider a latent structure space \mathcal{S} . It may be useful to view the space \mathcal{S} as a generalization of 3D point clouds like those used in standard 3D reconstruction. An element in this space generates images in image space \mathcal{I} via a projection operator $\pi : \mathcal{S} \rightarrow \mathcal{I}$. We also define a latent motion space \mathcal{M} , which is some subset of the set of homeomorphisms on \mathcal{S} .

For any element S of the structure space \mathcal{S} , a continuous motion sequence $\{M_t \in \mathcal{M} \mid t \geq 0\}$ generates a continuous image sequence $\{I_t \in \mathcal{I} \mid t \geq 0\}$ where $I_t = \pi(M_t(S))$. For a discrete set of images, we can rewrite this as $I_t = \pi((M_{\Delta_t} \circ M_{t-1})(S)) = \pi(M_{\Delta_t}(S_{t-1}))$, which defines a hidden Markov model, as illustrated in Figure 5 (a). As \mathcal{M} is a set of invertible functions, it automatically forms a group and thus has an identity element, unique inverses, and is associative under composition.

A simple example of this is the case of rigid image motion, such as that produced by a camera translating and rotating through a rigid scene in 3D; here, the latent structure of the scene \mathcal{S} is a point cloud, and the latent motion space is $\mathcal{M} = \mathcal{SE}(3)$. For a scene with N rigid bodies, we can describe the motion with a tuple of $\mathcal{SE}(3)$ values, i.e. $\mathcal{M} = (\mathcal{SE}(3))^N$, where the N th motion acts on the set of points belonging to the N th rigid body (Figure 5 (b)). For scenes that are not well modeled as a collection of rigid bodies, we can instead consider \mathcal{M} as the space of all homeomorphisms on the latent space. Each of these descriptions of motion (rigid scene, N rigid bodies, general nonrigid scenes under homeomorphisms) has the desired group properties of associativity and invertibility.

3.2. Learning motion by group properties

Our goal is to learn a function $\Phi : \mathcal{I} \times \mathcal{I} \rightarrow \overline{\mathcal{M}}$ that maps pairs of images to a representation $\overline{\mathcal{M}}$ of the motion space \mathcal{M} . For this to be useful, we also learn a corresponding composition operator $\diamond : \overline{\mathcal{M}} \rightarrow \overline{\mathcal{M}}$ that emulates the composition of elements in \mathcal{M} . This representation and operator should respect the group properties of motion.

It is possible in principle to learn such a representation in a supervised manner, given the pose of the camera and each independent object at every time step. However, supervised learning becomes infeasible in practice due to the difficulty of accurately labeling even rigid motion in real image sequences at scale, as well as to the nuisance factors of π and S , which become difficult to represent explicitly in more general settings.

Instead, we exploit the structure of the problem to learn to represent and compose without labels. If an image sequence $\{I_t\}$ is sampled from a continuous motion sequence, then the sequence representation should have the following properties for all t, t', t'' , reflecting the group properties of the latent motion:

$$(i) \text{ Associativity: } \Phi(I_{t-1}, I_t) \diamond \Phi(I_t, I_{t+1}) =$$

$\Phi(I_{t-1}, I_{t+1})$. The motion of composed subsequences is the total motion of the sequence.

- (ii) Existence of the identity element: $\Phi(I_t, I_t) \diamond \Phi(I_{t'}, I_{t''}) = e \diamond \Phi(I_{t'}, I_{t''}) = \Phi(I_{t'}, I_{t''})$. Null image motion corresponds to the (unique) identity in the latent.
- (iii) Invertibility: $\Phi(I_t, I_{t+1}) \diamond \Phi(I_{t+1}, I_t) = e$. The motion of the reversed images is the inverse of the motion of the images.

We recompose image sequences to learn a representation of the latent motion space with these properties. We use an embedding loss to approximately enforce associativity and invertibility among subsequences sampled from an image sequence. Associativity is encouraged by pushing differently composed sequences with equivalent motion to have same representation. Invertibility of the representation is encouraged by pushing all loops to the same representation (i.e. to a learned analogue of the identity in the embedding space). We encourage the uniqueness of the identity representation by pushing non-identity sequences and loops away from each other in the representation. To avoid learning a trivial representation, we push sequences with non-equivalent motion to different representations. This procedure is illustrated schematically in Figure 4 (a).

It is perhaps useful to compare the model framework used here to brightness constancy, the scene assumption that forms the basis of optical flow methods [23]. The general framework shown in Figure 5 (a) encompasses brightness constancy, which additionally constrains the projection of every world point to take the same luminance value in all frames (this is a constraint on π). Rather than relying on constraints on image-level properties, which are inappropriate in many settings, we learn a representation using assumptions about the group properties of motion in the world that are valid even over long temporal sequences.

Our method has several limitations. First, we assume relative stability in terms of the structural decomposition of the scene. Our method will have difficulty representing motion in cases of occlusion or “latent flicker”: i.e. if structural elements are not present in most frames. In particular, our method is not designed to handle cuts in a video sequence. Because we use a learned representation, we do not expect our representation to perform well on motions or scene structures that differ dramatically from the training data. Finally, the latent structure and motion of a scene are in general non-identifiable. This implies that for any given scene, there are several $\bar{\mathcal{M}}$ that can adequately represent \mathcal{M} . We do not claim to learn a unique representation of motion, but rather we attempt to capture one such representation.

3.3. Sequence learning with neural networks

The functions Φ and \diamond are implemented as a convolutional neural network (CNN) and a recurrent neural network (RNN), respectively, which allows them to be trained over image sequences of arbitrary length. For the RNN, we use the long short-term memory (LSTM) model [24] due to its ability to reliably process information over long time sequences. The input to the network is in an image sequence $[I_1, \dots, I_t]$. The CNN Φ processes these images and outputs an intermediate representation $[C_{1,2}, \dots, C_{t,t-1}]$. The LSTM operates over the sequence of CNN outputs to produce an embedding sequence $[R_{1,2}, \dots, R_{t-1,t}]$. We treat $R(\{I_t\}) = R_{t-1}$ as the embedding of sequence $\{I_t\}$. This configuration is illustrated schematically in Figure 2.

At training time, our network is structured as a Siamese network [25] where pairs of sequences are passed to two separate networks with shared weights. The network is trained to minimize a hinge loss with respect to the output embedding of these pairs:

$$L(R^1, R^2) = \begin{cases} d(R^1, R^2), & \text{if positive pair} \\ \max(0, m - d(R^1, R^2)), & \text{if negative pair} \end{cases} \quad (1)$$

where $d(R^1, R^2)$ measures the distance between the embeddings of two example sequences $\{I_{t_1}^1\}$ and $\{I_{t_2}^2\}$, and m is a fixed scalar margin (we use $m = 0.5$ for all experiments). Positive examples are image sequences that are compositionally equivalent, while negative examples are those that are not. We use the cosine distance as the distance between the two embeddings in all experiments reported here (we obtained similar results using the euclidean distance).

We include six recomposed subsequences for each training sequence included in a minibatch: two forward, two backward, and two identity subsequences, as shown in Figure 4 (a). Subsequences are sampled such that all three sequence types share a portion of their frames. If the sampling were done using only strictly overlapping sets of images, it is possible that the network could learn to distinguish sequences simply by comparing the last images. To prevent this, we use several image recomposing schemes (shown in Figure 4 (b)). Because the network is exposed to sequences with the same start and end frames but different motion, this sampling procedure encourages the network to rely on features in the motion domain, rather than purely on image-level differences.

We note also that learning the function Φ using pairs of images as input decreases the data efficiency relative to single-image input, as all images interior to a sequence are processed twice. We also explored learning a representation Φ taking single images as input. This formulation relies on a different view of the relationship between

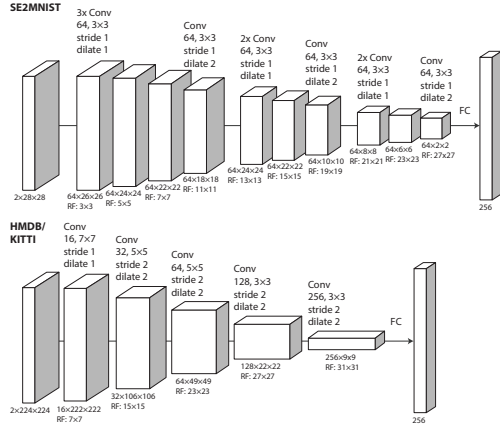


Figure 6: CNN architectures used in this paper, labeled by the datasets they were used on. The output of each CNN is fed to an LSTM in all cases. RF values indicate the receptive field size in pixels.

| Method | SE2MNIST | HMDB | KITTI |
|---------------------|----------|----------|----------|
| Chance | 1.0 | 1.0 | 1.0 |
| Image pairs, equiv | $8.1e-4$ | $6.3e-3$ | $7.2e-3$ |
| Image pairs, ineq | $1.7e-2$ | 0.11 | $8.0e-2$ |
| Single image, equiv | 0.74 | $4.5e-2$ | $3.5e-2$ |
| Single image, ineq | 0.79 | 3.3 | 3.5 |

Table 1: Average embedding error for group property on test data. Results are averaged over forward, backward, and loop sequences, comparing equivalent and inequivalent sequence compositions. The method taking two images as input can learn association and inversion in a wide variety of image contexts.

the structure/motion inferred by the representation and the CNN/RNN architectural division. Because the CNN in this configuration only has access to single images, it cannot extract local image motion directly. We found that the representation learned from image pairs outperformed the one learned from single images in all domains we tested (see Table 2).

4. Network implementation and training

In all experiments presented here, the networks were trained using Adam [26]. For SE2MNIST training, we used a fixed decay schedule of 30 epochs. $1e-2$ was a typical starting learning rate. We made batch sizes as large as possible while still fitting on GPU. For SE2MNIST, typical batch sizes were 50-60 sequences, whereas for HMDB [27] and KITTI [28] the batch sizes were typically 25-30 sequences. All networks were implemented in Torch 7 [29]. We used

| True middle frame | Best frame (all other classes) |
|-------------------|--------------------------------|
| 0.030 | 0.19 |

Table 2: We evaluate our method on nearest neighbor recall by sequence completion. Two nearby frames (A and C) from HMDB are used as a probe, and we compare the distance between $A \rightarrow B \rightarrow C$ and $A \rightarrow C$, where frame B is either drawn from between A and C or a frame from another class of HMDB. We compare the average error for the true middle frame to the average of minimum errors over all other classes. See the supplement for details.

code from [30] and [31] in our network implementation.

CNN architectures are shown in Figure 6. We use dilated convolutions throughout to obtain large output-unit receptive fields suitable for capturing spatially large-scale patterns of motion. We used ReLU nonlinearities and batch normalization [32] after each convolutional layer. In all cases, the output of the CNN was passed to an LSTM with 256 hidden units, followed by a final output linear layer with 256 hidden units. This architectural configuration is similar to the CNN-LSTM architectures used for action classification in [33] and [34]. In all experiments shown here, CNN-LSTMs were trained on sequences 3-5 images in length. We tested the SE2MNIST network with sequences of length up to 10 with similar results.

5. Experiments

We first demonstrate that our learning procedure can discover the structure of motion in the context of rigid scenes undergoing 2D translations and rotations. We then show results from our method on action sequences from the HMDB51 dataset [27]. Our procedure can learn a motion representation without requiring flow extraction in this context, and thus may be complimentary to standard representation derived from ImageNet pretraining. Finally, we show that our method learns features useful for representing independent motions on the KITTI tracking dataset [28].

5.1. Rigid motion in 2D

To test the ability of our learning procedure to represent motion, we trained a network on a dataset consisting of image sequences created from the MNIST dataset (called SE2MNIST). Each sequence consists of images undergoing a coherent, smooth motion drawn from the group of 2D, in-plane translations and rotations ($\mathcal{SE}(2)$). Each sequence was created by smoothly translating and rotating an image over the course of twenty frames. We sampled transformation parameters uniformly from $[-10, 10]$ pixels for both horizontal and vertical translation and from $[0, 360]$ degrees for rotation.

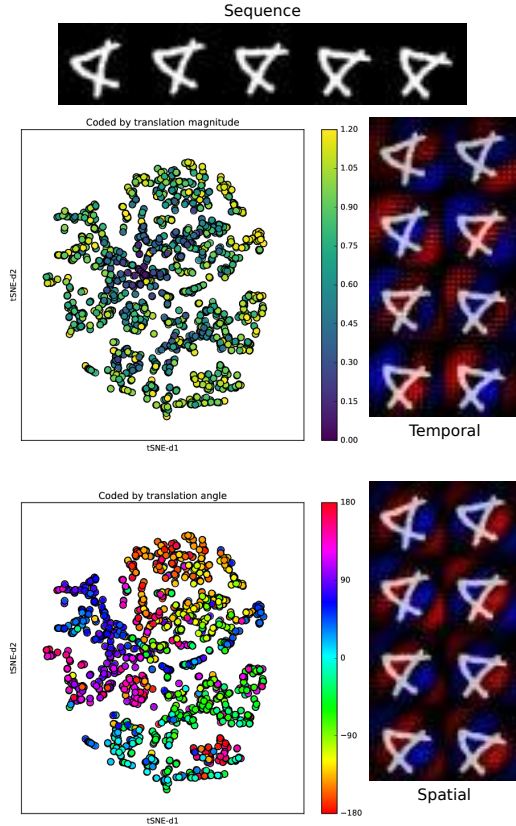


Figure 7: An example test sequence from SE2MNIST and the corresponding saliency maps (right). Spatial saliencies show the gradient backpropagated from the CNN, while temporal saliencies show the gradient backpropagated from the final RNN timestep. Each row represents an image pair passed to one of the CNNs. We also show tSNE of the network embedding on the test set, with points labeled by (top left) the magnitude of translation in pixels and (bottom left) the translation direction in degrees. The representation clearly clusters sequences by both translation magnitude and direction.

To probe the representation learned by the network on this data, we ran dimensionality reduction using tSNE [35] on the full sequence embedding (the output of the LSTM at the final timestep) for unseen test images undergoing a random, noiseless translation. The results are shown in Figure 7. The network representation clearly clusters sequences by both the direction and magnitude of translation. No obvious clusters appear in terms of the image content. This suggests that the network has learned a representation that captures the properties of motion in the training set. This content-invariant clustering occurs even though the network was never trained to compare images with different spatial

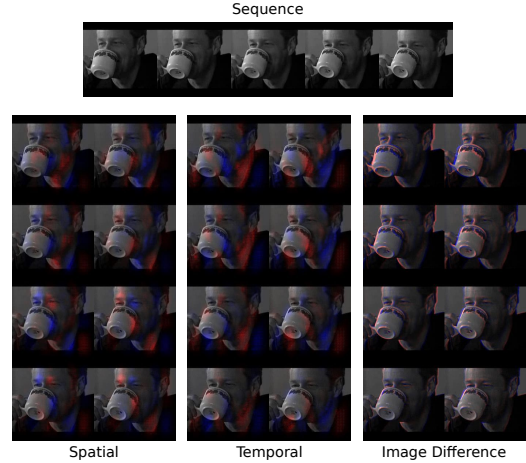


Figure 8: A typical saliency map for a sequence from the HMDB test set. Both spatial and temporal saliency maps highlight locations of actual and potential movement in the physical scene, while the difference image is only sensitive to pixel-level differences.

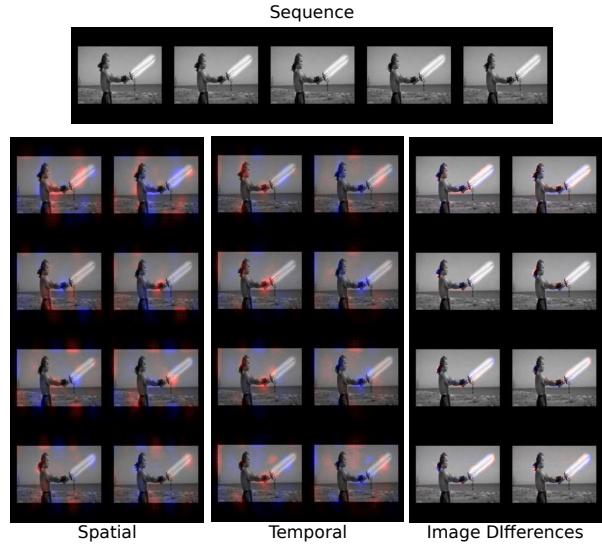


Figure 9: A sequence from HMDB with large embedding error, due to the limited motion in the sequence. Despite the small motion, the network still highlights areas likely to generate movement, such as the head and sword hilt.

content and the same motion.

To further probe the network, we visualize image-conditioned saliency maps in Figure 7. These saliency maps show the positive (red) and negative (blue) gradients of the network activation with respect to the input image (i.e. backpropagated to pixel space). As discussed in [36],

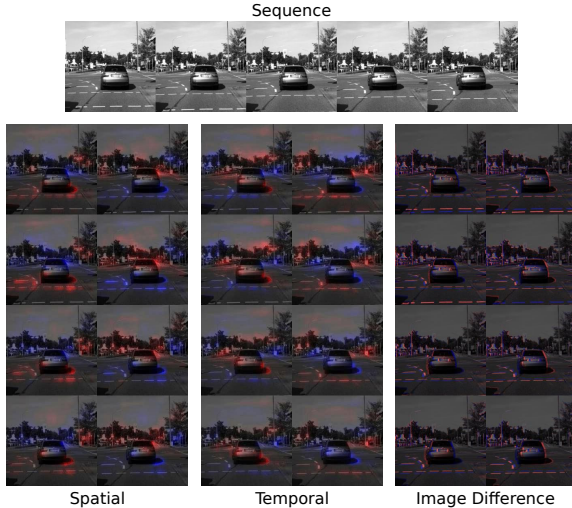


Figure 10: Saliency results on a motion sequence from KITTI tracking with both camera and independent motion.

such a saliency map can be interpreted as a first-order Taylor expansion of the function Φ , evaluated at image I . The saliency map thus gives an indication of how the pixel intensity values affect the network’s representation. These saliency maps show gradients with respect to the output of the CNN over a pair of images (spatial) or with respect to the output of the LSTM over the sequence (temporal).

Intriguingly, the saliency maps at any given image pair bear a strong resemblance to the spatiotemporal energy filters of classical motion processing [37] (and which are known to be optimal for 2D speed estimation in natural scenes under certain assumptions [38]). We note that these saliency maps do not simply depict the shape of the filters of the first layers, which more closely resemble point detectors, but rather represent the implicit filter instantiated by the full network on this image. When compared across different frames, it becomes clear that the functional mapping learned by the network can flexibly adapt in orientation and arrangement to the image content, unlike standard energy model filters.

5.2. Human Actions

To demonstrate that our method can learn to represent motion in a real-world setting, we train it on action recognition sequences from HMDB51 [27], a standard dataset of image sequences depicting a variety of human actions.

We show an example of the resulting saliency maps in Figure 8. Note in this example that our network highlights parts of the image relevant to motion, such as the edge of the cup, the eyebrows, and the neck. In contrast, taking image differences simply highlights image edges that move lo-

cally and does not reflect information about the scene structure, such as longer-term changes or the tendency of some body parts to move. The saliency maps consistently display a contrast between the left and right images of a pair, which suggests that the highlighted regions display where motion is detected as well as where the affected regions should move.

These saliency patterns are typical for sequences in the dataset, including sequences with large embedding loss. The embedding loss is largest in cases where the assumptions of our method are violated. This includes sequences with no motion, with cuts (which introduce dramatic structure changes and non-continuous motion), and to a lesser extent, sequences where motion is obscured by occlusions. Even in these examples, saliency maps highlight image regions that are pertinent for motion analysis, even if no motion is actually present in the sequence (Figure 9).

5.3. Independent Motion

Finally, we show results from our method in the presence of independent motion. We use the KITTI object tracking dataset [28]. Here we use the saliency maps of our network to show that the representation highlights the motion of independent objects over time. In Figure 10 we show the saliency on some example frames in the dataset. Note that the saliency map highlights objects moving in the background, such as the trees, poles, and road markers, and the independent motions of the car in the foreground. The network highlights the moving areas of the car, such as the bumper, even when these areas don’t contain strong image gradients. This effect is particularly noticeable in the saliencies generated by the temporal representation, which integrates motion over a longer temporal window. See the supplement for results on longer sequences featuring independent motions. These results suggest our method may be useful for independent motion detection and tracking.

6. Conclusion

We have presented a new method for unsupervised learning of motion representations. We have shown results suggesting that enforcing group properties of motion is sufficient to learn a representation of image motion. The results presented here suggest that this representation is able to generalize between scenes with disparate scene and motion content to learn a motion state representation that may be useful for navigation and other behavioral tasks relying on motion. Because of the wide availability of long, unlabeled video sequences in many settings, we expect our approach to be useful for generating better global motion representations in a wide variety of real-world tasks.

References

- [1] M. B. Reiser and M. H. Dickinson, "Visual motion speed determines a behavioral switch from forward flight to expansion avoidance in *Drosophila*," *The Journal of Experimental Biology*, vol. 216, pp. 719–732, Feb. 2013.
- [2] D. J. Heeger and A. D. Jepson, "Subspace methods for recovering rigid motion i: Algorithm and implementation," *International Journal of Computer Vision*, vol. 7, no. 2, pp. 95–117, 1992.
- [3] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE Robotics & Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.
- [4] F. Fraundorfer and D. Scaramuzza, "Visual odometry: Part ii: Matching, robustness, optimization, and applications," *IEEE Robotics & Automation Magazine*, vol. 19, no. 2, pp. 78–90, 2012.
- [5] D. Nistér, "Preemptive ransac for live structure and motion estimation," *Machine Vision and Applications*, vol. 16, no. 5, pp. 321–329, 2005.
- [6] A. Jaegle, S. Phillips, and K. Daniilidis, "Fast, robust, continuous monocular egomotion computation," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 773–780, May 2016.
- [7] D. Sun, S. Roth, and M. J. Black, "Secrets of optical flow estimation and their principles," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 2432–2439, IEEE, 2010.
- [8] A. Wedel, C. Rabe, T. Vaudrey, T. Brox, U. Franke, and D. Cremers, "Efficient dense scene flow from sparse or dense stereo data," in *European Conference on Computer Vision*, pp. 739–751, Springer, 2008.
- [9] J. Xiao, J.-x. Chai, and T. Kanade, "A closed-form solution to non-rigid shape and motion recovery," in *European Conference on Computer Vision*, pp. 573–587, Springer, 2004.
- [10] I. Laptev, "On space-time interest points," *International Journal of Computer Vision*, vol. 64, no. 2-3, pp. 107–123, 2005.
- [11] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 4489–4497, IEEE, 2015.
- [12] Q. V. Le, "Building high-level features using large scale unsupervised learning," in *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 8595–8598, IEEE, 2013.
- [13] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1422–1430, 2015.
- [14] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," *arXiv preprint arXiv:1603.09246*, 2016.
- [15] J. J. Yu, A. W. Harley, and K. G. Derpanis, "Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness," *arXiv preprint arXiv:1608.05842*, 2016.
- [16] V. Patraucean, A. Handa, and R. Cipolla, "Spatio-temporal video autoencoder with differentiable memory,"
- [17] R. Garg and I. Reid, "Unsupervised cnn for single view depth estimation: Geometry to the rescue," *arXiv preprint arXiv:1603.04992*, 2016.
- [18] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2794–2802, 2015.
- [19] L. Wiskott and T. J. Sejnowski, "Slow feature analysis: Unsupervised learning of invariances," *Neural computation*, vol. 14, no. 4, pp. 715–770, 2002.
- [20] I. Misra, C. L. Zitnick, and M. Hebert, "Shuffle and learn: unsupervised learning using temporal order verification," in *European Conference on Computer Vision*, pp. 527–544, Springer, 2016.
- [21] D. Jayaraman and K. Grauman, "Learning image representations tied to ego-motion," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1413–1421, 2015.
- [22] P. Agrawal, J. Carreira, and J. Malik, "Learning to see by moving," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 37–45, 2015.
- [23] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, no. 1, pp. 185 – 203, 1981.
- [24] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [25] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 539–546, IEEE, 2005.
- [26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.
- [27] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: a large video database for human motion recognition," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011.
- [28] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
- [29] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *BigLearn, NIPS Workshop*, 2011.
- [30] J. Johnson, "torch-rnn." <https://github.com/jcjohnson/torch-rnn>, 2016.

- [31] G. Thung, “torch-lrcn.” <https://github.com/garythung/torch-lrcn>, 2016.
- [32] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,”
- [33] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, “Long-term recurrent convolutional networks for visual recognition and description,” pp. 2625–2634.
- [34] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, “Beyond short snippets: Deep networks for video classification,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4694–4702.
- [35] L. Van der Maaten and G. Hinton, “Visualizing data using t-SNE,” vol. 9, no. 2579, p. 85.
- [36] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep inside convolutional networks: Visualising image classification models and saliency maps,”
- [37] E. H. Adelson and J. R. Bergen, “Spatiotemporal energy models for the perception of motion,” vol. 2, no. 2, pp. 284–299.
- [38] J. Burge and W. S. Geisler, “Optimal speed estimation in natural image movies predicts human performance,” vol. 6, p. 7900.

Supplement: Unsupervised learning of image motion by recomposing sequences

Andrew Jaegle*, Stephen Phillips*, Daphne Ippolito, and Kostas Daniilidis
University of Pennsylvania
Philadelphia, PA USA

{ajaegle, stephi, daphnei, kostas}@seas.upenn.edu
















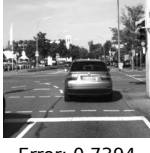


| Reference Images | True Image | In Sequence, Outside Range | | Outside Sequence | |
|--|---|--|---|---|---|
| | | Embedding | Euclidean | Embedding | Euclidean |
|  Skip: 1 |  Error: 1.000 |  Error: 35.90 |  Error: 0.9584 |  Error: 214.9 |  Error: 2.098 |
|  Skip: 2 |  Error: 1.000 |  Error: 14.96 |  Error: 0.7121 |  Error: 215.2 |  Error: 1.487 |
|  Skip: 3 |  Error: 1.000 |  Error: 0.4809 |  Error: 0.7394 |  Error: 41.85 |  Error: 1.427 |

Figure 1: Examples nearest neighbors, as in Table 1. Errors here are all relative to the error of the test images, for both embedding and Euclidean distances. Note that for skip-1 and skip-2 cases, the embedding distance for the true frame is dramatically lower than for all other frames.

1. Nearest Neighbor Experiments

To test the smoothness of our learned representation and its usefulness for localization, we ran a nearest neighbor experiment. For a given sequence, we select a query start frame (frame A), a query end frame (frame C), and a frame between them (frame B). The spacing between frames A and B and frames B and C is equal. For HMDB, we choose a skip of 1 frame. We find the embedding of the sequences $A \rightarrow B \rightarrow C$ and $A \rightarrow C$. We then choose alternate frames (B') from test sequences of HMDB and compute the embedding $A \rightarrow B' \rightarrow C$. To find the nearest neighbors to query

frames A and C, we compare the distance of the embedding of $A \rightarrow C$ from the embeddings $A \rightarrow B' \rightarrow C$ for all choices of B' . If the sequence embedding is smooth and has learned the temporal properties of sequences on the dataset, this distance should be lowest for the true middle frame. We show the results on HMDB in Table 1 in the main paper. We compare the average embedding distance of the true frame over all tested sequences to the minimum of the embedding distances of all other choices of B' . We sampled B' from 93 different sequences.

The results of a similar experiment on KITTI are shown

Table 1: Results of nearest neighbors comparisons on the KITTI dataset.

| | True Frame | Min. Seq. | Min. Out |
|---------------|------------|-----------|----------|
| Skip 1 | 3.91e-03 | 7.67e-02 | 2.94e-01 |
| Skip 2 | 1.18e-02 | 2.02e-02 | 1.34e-01 |
| Skip 3 | 1.82e-02 | 1.34e-02 | 8.65e-02 |
| Skip 1 (Euc.) | 7.92e-04 | 7.97e-04 | 1.09e-03 |
| Skip 2 (Euc.) | 9.59e-04 | 8.13e-04 | 1.08e-03 |
| Skip 3 (Euc.) | 1.05e-03 | 7.83e-04 | 1.07e-03 |

in Supplementary Table 1. Here, we compare the embedding produced by our procedure to an average pixel-wise Euclidean distance. As with HMDB, we use three successive frames and find the distance to the middle one (B) using the first (A) and last (C) frames. The skip value (1,2,3) gives the number of frames skipped between neighboring images. We compare embedding distances to the average pixel-wise Euclidean distance to frame A. We compare the embedding/Euclidean error on the true middle frame, the lowest error over all other frames in the same sequence, and the lowest error on randomly selected frames from other sequences. We sampled 20 random frames from each of the other KITTI test sequences here. Our method generally leads to embeddings that are smallest for the true middle frame, even when compared to other frames taken from the same sequence. Our network embedding has a smaller advantage in the skip-3 case: this is not surprising, as skips of 3 frames or greater were not seen during training. In all cases, the difference between the embedding error for the true middle frame and for unrelated images from KITTI is consistently smaller than the difference for Euclidean distances. We display sample nearest neighbor results obtained using this method in Supplementary Figure 1. In this figure, all distances are normalized to the distance on the true image, for both embedding and Euclidean distances.

2. Embedding error and the motion model

In Supplementary Figure 2, we show the embedding error for five-frame subsequences taken from all sequences on the HMDB test set. While the majority of embedding errors are quite small, a few sequences have larger errors. We show the sequences with the lowest and highest embedding errors in Supplementary Figure 3. The embedding error on test data is highest on data where the assumptions of the motion model are violated. The sequences with highest embedding error generally have no motion or fall directly on a cut (corresponding to a dramatic change in image content). Oscillatory motion, which does not violate the assumptions of the motion model, can sometimes make certain training comparisons inappropriate (as in the forward vs. loop example shown here).

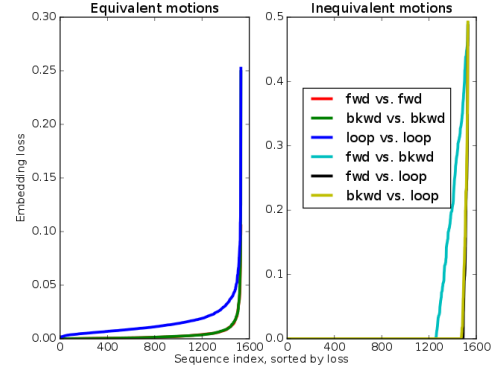


Figure 2: Embedding loss on test sequences. We show the loss on one randomly chosen subsequence from each of the sequences of the HMDB test set. Losses are split into contribution from the different types of equivalent and inequivalent sequences (see the main text for details).

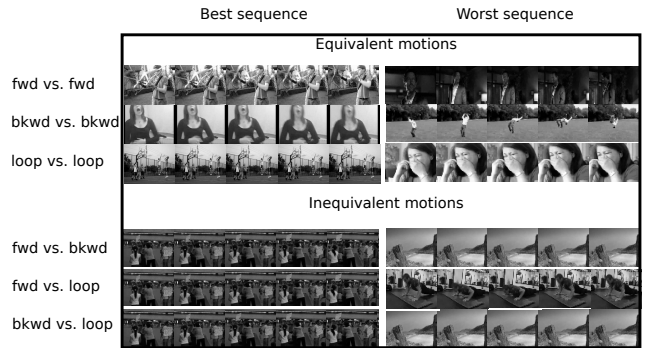


Figure 3: Subsequences with the lowest and highest embedding losses for each of the comparison types (shown in the columns labeled best and worst sequences, respectively). The worst sequences feature very little image motion, cuts, or oscillation.

3. Additional visualization on KITTI

We show results from our network on a longer sequence featuring independent motion in Supplementary Figure 4. Frames were obtained by taking crops from the first sequence of KITTI tracking. We randomly chose crops of dimension 224x224 containing the bounding box around the bike for all frames where the bike is present. We show spatial saliency maps (backpropagated from the output of the CNN), as in the main text. We show the saliency map for only the first frame of each input pair to the CNN for visualization purposes. The bike is highlighted prominently in the majority of frames, regardless of distance from the camera. When the cyclist is closer to the camera, the moving parts of

the bicycle (such as the wheels) are highlighted. These results suggest the usefulness of our method for independent motion analysis.



Figure 4: Spatial saliency maps for a longer sequence from KITTI including camera and independent motion. Frames are shown in temporal order, with the first frame at the top left and the last frame at the bottom right. For clarity of visualization, the saliency map is shown only for the first frame input to the network at each timestep.